TAGESRÜCKBLICK

Patrick Hettich

Patrick Hettich

06.11.2020

Allgemeines

Der Tag war sehr erfolgreich, es wurde ein Auftrag gegeben auf eine API zuzugreifen und diese darzustellen.
Wir haben als erstes ein Team gebildet und besprochen was für eine API und was für Technologien wir verwenden.

Schlussendlich wollen wir eine API verwenden, um Radiostationen zu erhalten und dann die jeweilig ausgewählte Station abspielen zu lassen.

Als Technologie verwenden wir Android-Studio mit Flutter/Dart um eine Applikation zu schreiben, welche dann theoretisch auf allen Geräten läuft, sowie die API von Laut.fm.

Ich freue mich auf das Modul, ein eigenes Projekt im Team zu entwickeln wird enorm spannend und hilfreich für das Arbeitsleben.

Zudem bin ich sehr zufrieden mit der Idee der Applikation, sowie den verwendeten Technologien.

Erfolge und Erledigtes

- Kick-off-Meeting
- API Auswahl: api.laut.fm
- Technologie: Android-Studio mit Flutter/Dart
- Erste Designs
- Erste Modell für die JSON-Responses
- Use-Case-Diagramm
- Erste Aufgaben wurden in DevOps erstellt.
- Git-Repository wurde erstellt und Collaborator hinzugefügt.

Aufgaben und Gedanken für das nächste Mal

- Ordnerstruktur der Applikation.
- Modelle der JSON-Responses fertigstellen.
- Provider-Klassen erstellen
    - Für Radio-Stationen
    - Für Lokale Dateien
- Mögliche Screen-Widgets planen
- Mögliche Unter-Widgets planen
- Walking-Skeleton erstellen und auf GitHub laden.
- Anhand von Skeleton erste Programmier-Aufgaben verteilen.

13.11.2020

Allgemeines

Zu Beginn ging das ganze wegen technischer Manko von Teams, DevOps und Android Studio etwas stockend von der Hand.
Sobald es aber mit dem morgen-Meeting los ging, wurden die Aufgaben schnell verteilt und jeder konnte mit den zugewiesenen Aufgaben beginnen.

Ich begann mit dem Walking-Skeleton für Models, Provider und Helper, sowie der allgemeinen Ordnerstruktur, danach stellte ich die Verbindung zur API her und holte von dort die Daten, was wegen eines kleinen Fehlers im Code, länger dauerte wie geplant.
Zuletzt wurden die Stationen in einer Liste dargestellt und via Player-Helper konnten die dann angeklickten Sender abgespielt werden.

Alles in allem ein sehr erfolgreicher Tag, das ganze Team war sehr konzentriert bei der Arbeit und es wurden schon jetzt gute Fortschritte erzielt.

Erfolge und Erledigtes

- Ordnerstruktur der Applikation.
- Modelle der JSON-Responses fertig.
- Provider-Klassen erstellt
    - Für Radio Stationen und Genres.
- Helper für lokale Dateien und Player wurde erstellt.
    - Zuerst war geplant ein Provider für lokale Dateien zu erstellen, was aber (bis jetzt) nicht notwendig ist.
- Screen-Widget für Radio-Stationen wurde erstellt.
- Walking-Skeleton erstellt.

Aufgaben und Gedanken für das nächste Mal

- Widget-Screen für die momentan laufende Station.
    - Grossanzeige der Radio-Station
    - Anzeige von laufendem Lied
- Innerhalb der Sender-Liste eine Möglichkeit zum Anhalten der Laufenden Station.
- Beginn mit Themes
    - Ordner und leere Files wurden bereits erstellt.
- Favoriten
    - Speichern lokal
    - Anzeigen von nur Favoriten in der Liste der Stationen
- Settings
    - Momentan nur Theme, Speichern und dann entsprechend laden.

20.11.2020

Allgemeines

Mein Hauptauftrag war das implementieren der zwei Farb-Themen, zuerst wurde ein File erstellt, welches diese beinhaltet, dies ging ohne weitere Probleme.

Als nächstes wurde im Sidedrawer ein Switch-Button eingefügt welche zwischen Dark- und Lightmode wechseln kann, um dies zu erreichen musste der vorherige Shared-Preference-Helper in einen Provider umgewandelt werden, der Grund dafür ist, dass beim Ändern über den Switch das Theme aktualisiert werden muss, da ansonsten das Programm neu gestartet werden muss um die Änderungen zu sehen.

Das Erstellen dieses Providers war keine allzu grosse Sache, da ein Grossteil des Helpers komplett übernommen werden konnte und nur mit NotifyListener() erweitert werden musste, um die Widgets, welch aktualisiert werden müssen zu benachrichtigen.
Jedoch entstand mit dem benutzen des Providers ein Problem, das Theme wird in der MaterialApp definiert, aber es ist unmöglich dort den Provider zu benutzen, da er auch dort eingebunden wird. Nach langem Suchen, war die Lösung einfach, die MaterialApp wird aus dem main-File genommen, in einem neuen File erstellt und dann eingebunden.

Die nächste Aufgabe war das Erstellen eines Widgets, welches in der Station-Liste liegt, und für das stoppen der momentanen spielenden Radio-Station verantwortlich ist, dies ist momentan relativ simpel aufgebaut und wird noch verschönert.

Zu guter Letzt habe ich noch den Station-Provider so erweitert, sodass die momentan spielende Station abgerufen werden kann, dadurch ist es simpler Informationen, welche in verschiedenen Widgets gebraucht werden, ohne props durch mehrere Widgets zu schicken, abzurufen.

Erfolge und Erledigtes

- Innerhalb der Sender-Liste eine Möglichkeit zum Anhalten der Laufenden Station.
    - Einfaches Widget erstellt, sollte noch verschönert werden.
- Themes
    - Dark- und Lightmode kann gewählt werden
    - Änderungen werden gespeichert
    - Bei Ändern des Modes sind Änderungen sofort sichtbar

Aufgaben und Gedanken für das nächste Mal

- Widget-Screen für die momentan laufende Station.
    - Grossanzeige der Radio-Station
    - Anzeige von laufendem Lied
- Favoriten
    - Speichern lokal
    - Anzeigen von nur Favoriten in der Liste der Stationen
- Suchfunktion
    - Es soll nach Stationen gesucht werden können.
    - Es soll nach Genres gesucht werden können.

27.11.2020

Am Morgen bestand meine Hauptaufgabe darin, Unterstützung beim Herstellen der Such- und Favoriten-Funktion.
Beides wurde mithilfe des Stations- und PreferencesProvider gelöst, was dazu führte, dass nicht extra ein weiteres Widget erstellt werden musste.

Am Nachmittag begann ich das bereits erstellte Widget für die einzelne Station zu verbessern, die Logik wurde bereits von Pascal erstellt, das Styling wurde mir überlassen, ich folgte dabei der erstellten Vorlage von Sandro.
Um das Abspielen in diesem Widget zu stoppen, musste ich den PlayerHelper in einen Provider abändern, um die Steuerung in mehreren Widgets mit der gleichen Klasse zu ermöglichen.

Vor Schluss wurden die Branches noch zusammen ge-merged, sowie entstandene Konflikte gelöst.

Erfolge und Erledigtes

- Suchfunktion
    - Es kann nach Stationen gesucht werden.
- Favoriten
    - Es können Stationen als Favoriten gespeichert werden.
    - Es wird eine Liste von Favoriten ausgegeben.
- Widget-Screen für die momentan laufende Station.
    - Grossanzeige der Radio-Station
    - Anzeige von laufendem Lied
    - Anzeigen der Länge und momentaner Position des Liedes
    - Speichern der Station als Favorit
- Suchfunktion
    - Es kann nach Stationen gesucht werden.

Aufgaben und Gedanken für das nächste Mal

- Readme erstellen!
- Favoriten
    - Unterschiedliches Icon von momentaner Station, falls bereits Favorit.
- Genres
    - Es soll eine Liste von Genres gezeigt werden, beim Klick auf einen Eintrag werden alle Stationen dieses Genres angezeigt.
    - Eintrag für alle Genres erstellen
    - Neue Abfrage an API verhindern bei öffnen von Genres
- Web-View
    - Das styling der Webseite soll im Vergleich zur Mobile-Applikation andres aufgebaut werden.
    - Immer offene Sidebar
    - Kein Wechsel in neues Fenster bei hören einer Station
    - Play-Footer für momentane Station mit Informationen erstellen
    - Scrollbar entwerfen

- Refactoring
    - Sidebar schliessen bei Klick auf Eintrag
    - Suchfunktion erweitern
        - Nach Station und Beschreibung suchen
        - Automatisch in Suchfeld bei Klick auf Lupe
        - Es soll nach Genres gesucht werden können.
    - Play-Footer erweitern
        - Momentanes Lied
        - Länge des Liedes
        - Stationsname
        - Nicht schliessen bei Stoppen von Lied
    - FutureBuilder so erweitern, dass alle Provider vor Beginn der Applikation initialisiert werden.
    - Back-To-Top Button erstellen
    - Designfarben eventuell verschönern